QT Engine

Introduction

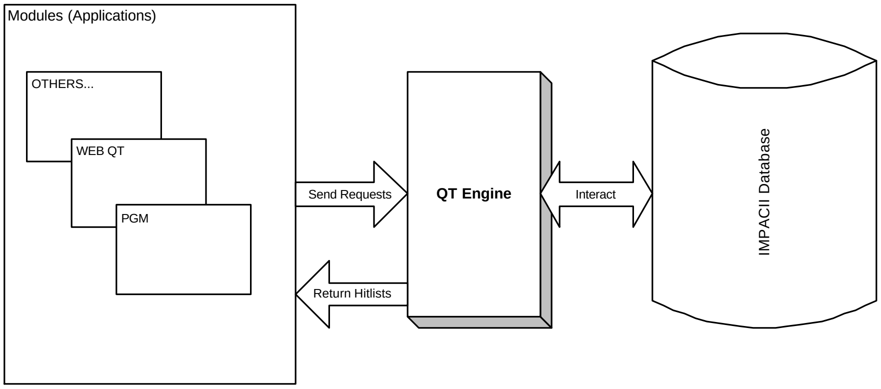
IMPACII and eRA Applications display various hit lists to end-users. Entities displayed in these hit lists include (but not limited to):

- Grant Applications (Web QT, Peer Review, ICO, PGM, etc.)
- People (Person Search and Administration Screens, etc.)
- Meetings (Committee Management, Peer Review, IAR, etc.)
- Institutions (IPF, Commons, etc.)

Cross cutting needs:

- The way to build the logic to execute searches and the way to display information to the end users needs to be standardized to enable future ease of maintenance and enhancement implementations.
- Given the complexity of the enterprise OLTP database, the learning curve for Application Developers is very steep. Meta layer to map the fields that represent different entities and their relationships needs to be developed.
- Further customization of the hit lists and queries by end-users needs to be supported.
- Further customization of sort order of records by end-users on the hit list needs to be supported.

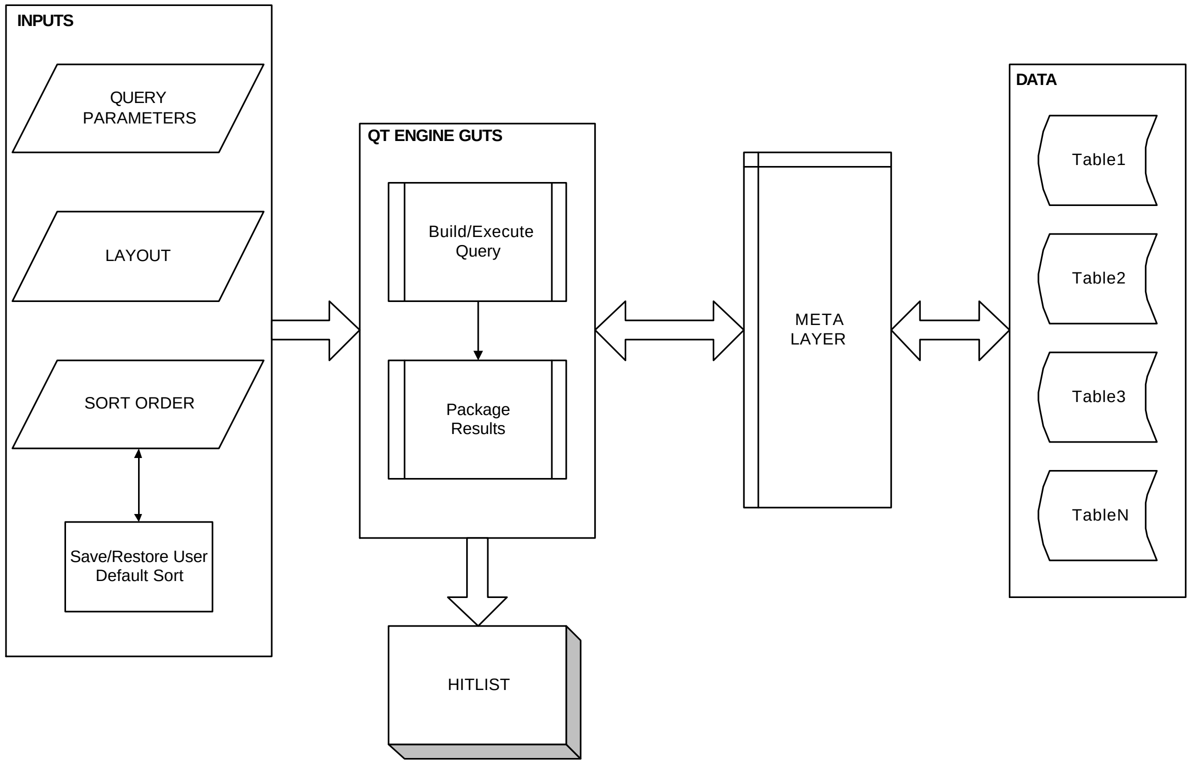
Following diagram explains how QT Engine fits into the overall project:



Purpose

1. Support Query Capabilities of canned (pre-defined) and Ad Hoc (user Entered) queries.
2. Support on the fly and/or pre-defined simple (column level) and nested (multicolumn) sorts.
3. Support ability of end users to define and save a user specific sort order per hit list/query.
4. Provide a uniform way to draw data from various tables via the Meta layer.
5. Lay ground for future end-user and Application developer’s abilities to customize query (hit list) results and searches (queries).
6. Ability to Store/Access pre-defined query.
7. Ability to accept query parameters from the calling program.
8. Prevent open-ended queries by checking for the required fields of the calling module (for Ad Hoc queries only).

Design



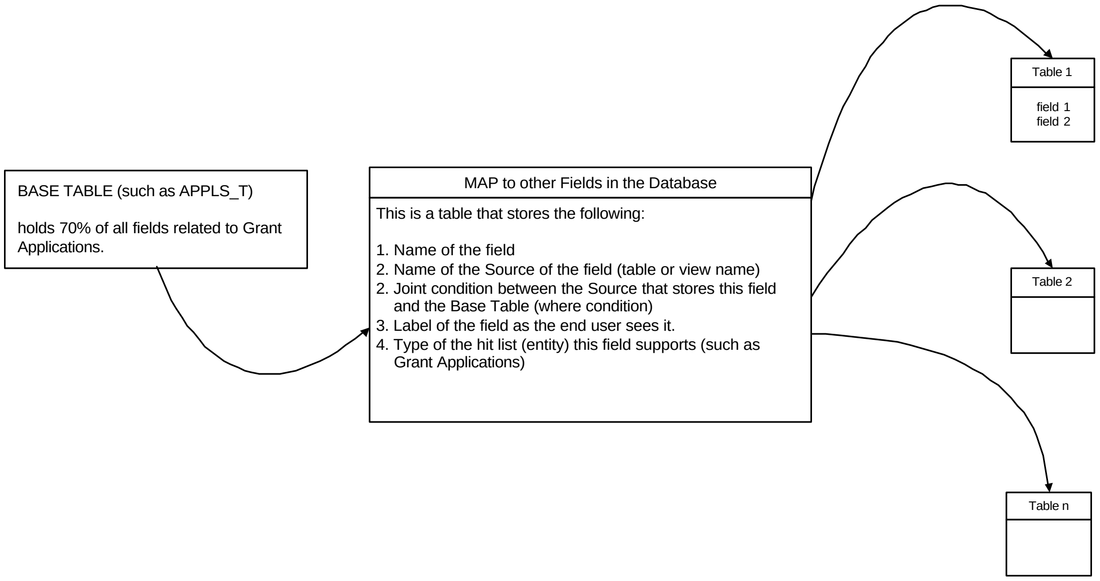
1.1 Meta Layer

To support a uniform way to build SELECT statements throughout the enterprise as well as to abstract Application Developers from the tedious tasks of writing individual SELECT statements themselves, the Meta layer to data stored in various enterprise database tables is proposed.

The Meta layer is entity-centric. It supports “entities” that exist in real life, such as grant applications or institutions, etc. This design specifically supports grant applications “types” of queries, but is generic to support other entities as requirements are formulated in the future.

Central table to support the persistence of grant applications is APPLS_T table (Base Table). It stores about 70% of attributes (fields) related to grant applications.

Meta Layer is a “map” of all other fields that make up the grant application information to the base table. The following figure illustrates the concept:


BASE TABLE (such as APPLS_T)

holds 70% of all fields related to Grant Applications.

MAP to other Fields in the Database

This is a table that stores the following:

1. Name of the field
2. Name of the Source of the field (table or view name)
2. Joint condition between the Source that stores this field and the Base Table (where condition)
3. Label of the field as the end user sees it.
4. Type of the hit list (entity) this field supports (such as Grant Applications)